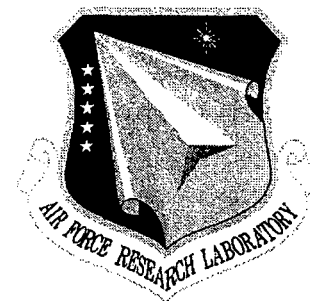


AFRL-IF-RS-TR-2002-125
Final Technical Report
June 2002



HIGH-PERFORMANCE KNOWLEDGE BASE SUPPORT FOR MONITORING, ANALYSIS, AND INTERPRETATION TASKS

Massachusetts Institute of Technology

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. F108

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

20020805 087

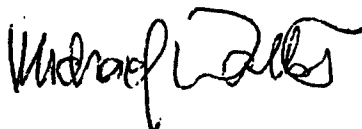
AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2002-125 has been reviewed and is approved for publication.



APPROVED: JOHN SPINA
Project Engineer



FOR THE DIRECTOR: MICHAEL L. TALBERT, Technical Advisor .
Information Technology Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFTD, 525 Brooks Road, Rome, NY 13441-4505. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE Jun 02		3. REPORT TYPE AND DATES COVERED Final May 97 - May 01
4. TITLE AND SUBTITLE HIGH-PERFORMANCE KNOWLEDGE BASE SUPPORT FOR MONITORING, ANALYSIS, AND INTERPRETATION TASKS			5. FUNDING NUMBERS C - F30602-97-1-0193 PE - 62301E PR - IIST TA - 00 WU - 08	
6. AUTHOR(S) Jon Doyle and Peter Szolovits				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Massachusetts Institute of Technology 545 Technology Square Cambridge, MA 02139			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/IFTD 525 Brooks Road Rome, NY 13441-4505			10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2002-125	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: John Spina, IFTD, 315-330-4032				
12a. DISTRIBUTION AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) The Monitoring, Analysis, and Interpretation Tool Arsenal (MAITA) is a knowledge-based tool that helps reduce time and effort of constructing a monitoring system at the start and of modifying an operational system of monitors to address special and temporary concerns. The architecture provides a rich library of monitoring systems enabling easy composition, modification, testing and abstraction of these library elements.				
14. SUBJECT TERMS knowledge base, monitoring, high-performance computing			15. NUMBER OF PAGES 40	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

Contents

1	Motivation and aims	1
1.1	Overview	1
1.1.1	Distributed monitoring	2
1.1.2	Stable monitoring	2
1.1.3	Secure monitoring	2
1.1.4	Open monitoring	3
1.1.5	Intelligent monitoring	3
1.1.6	Evolutionary monitoring	3
2	Monitoring concepts	5
2.1	Information flows	5
2.1.1	Signals	5
2.1.2	Alerts	5
2.1.3	Local stores and memories	5
2.2	Event recognition	6
2.2.1	Transducers	6
2.2.2	Correlators	6
2.2.3	Trend templates	6
2.2.4	Matching methods	7
2.2.5	Recognition control	7
2.3	Event reporting	8
2.3.1	Alerting control	8
2.3.2	Displays	9
2.4	Monitoring knowledge	9
2.4.1	Libraries	9
2.4.2	Ontology	11
2.4.3	Editing tools	11
3	Demonstration	13
3.1	Logging in	14
3.2	Creating a monitoring network	14
3.3	Locating a library process	15
3.4	Examining the new network	15
3.5	Navigating the network display	16
3.6	Starting the monitor	16
3.7	Displaying information flows	18
3.8	Strip charts and text alerts	18

3.9	Two-dimensional plots	18
3.10	Controlling processing	19
3.11	Quitting processes and logging off	19
4	Applications	21
4.1	Battlefield movement analysis	21
4.2	Information assurance	22
4.3	Additional developments	23
5	Conclusion	24
A	List of personnel	27
B	List of contract publications	28

Abstract: The *Monitoring, Analysis, and Interpretation Tool Arsenal* (MAITA) project was initiated to develop knowledge-based tools for constructing monitoring systems aimed at reducing the costs—in time, effort and expertise—of constructing a monitoring system at the start and of modifying an operational system of monitors to address special and temporary concerns. The MAITA architecture thus provides for a rich library of monitoring systems and mechanisms enabling easy composition, modification, testing and abstraction of these library elements.

The MAITA system was developed concurrently with explorations of applications to knowledge-based battlefield movement and computer security monitoring. In the initial battlefield application, MAITA monitoring processes successfully detected convoys and other battlefield movements in simulated airborne doppler radar data. The later computer security application was in preliminary stages at the end of the grant period but has subsequently detected intrusions and other events in simulated network data.

Chapter 1

Motivation and aims

We designed the MAITA system to help address two difficult problems: monitoring in context, and monitoring of special or temporary concerns.

Battlefield movement monitoring provides good examples of the problem of monitoring in context. A battlefield may have many movements ongoing simultaneously. Some of these may appear benign or routine in isolation although anything but when considered in the context of other movements. For example, one routinely sees small convoys bringing food, fuel, and munitions to forward deployments from supply depots. SCUD missile launcher convoys may have similar size and appearance given the limited resolution of long-distance doppler radar systems. Routine supply movements offer no reason for concern; indeed, their absence or halting might indicate more concern. SCUD convoy displacements always deserve close attention, as they represent key targets. One distinguishes the two convoy types more by contextual differences than by any intrinsic properties of their movements. SCUD convoy movements have correlation with thermal blooms indicating missile launches and rendezvous with two-truck resupply convoys emanating from missile depots. Regular forward supply movements lack such correlations and typically exhibit much greater correlation with the daily clock.

Computer security provides good examples of the problem of special or temporary concerns. Miscreants attempting to break into one site often repeat their attacks at other sites. A successful or partially successful attack against one site may call for alerting similar or related sites of the special characteristics of the attack, since the attack may have involved several events which mean little in themselves but which in retrospect appeared essential elements of the successful attack. For example, a random request from an obscure Lilliputian site might immediately precede a seemingly unrelated request from a Blefescucian site in the course of the attack on a particular service. If the succumbing site can alert its neighbors to this fact, the neighbors may repel similar attacks by watching for this specific combination and taking appropriate amelioratory actions pending a longer term solution to the vulnerability. The problem here is how to rapidly modify monitoring systems in place to recognize additional or specialized conditions and how to easily remove these specialized additions as windows of opportunity close or shift.

1.1 Overview

The MAITA system provides tools both for constructing intelligent monitoring networks that exploit domain knowledge to sift and correlate source-level signals, and for rapidly modifying running networks to address specialized or temporary concerns.

Chapter 2 sketches the main monitoring concepts involved in the MAITA system. The attached

paper *The Architecture of MAITA* provides a detailed description of the MAITA system architecture. Chapter 3 illustrates one version of the system in operation. Chapter 4 describes applications of MAITA to battlefield and computer security tasks.

This document does not describe the MAITA language for describing temporal signals, patterns, events, and monitoring processes, which is still under development. The remainder of this overview outlines the motivations for the overall system organization.

1.1.1 Distributed monitoring

The central concept in the MAITA architecture is that of a network of distributed monitoring processes. The metaphor we use for thinking of the operation of these monitoring networks is that of electrical networks, in which we "wire together" various components and network fragments by connecting their terminals together. In the computational context, individual monitoring processes take the place of electrical components, and transmitting streams of reports takes the place of electrical conduction. The set of monitoring processes form the nodes of the network, and the communication paths form the edges or links of the network. Each process in the network may have a number of "terminals", each of which receives or emits streams of reports. The network may exhibit a hierarchical structure, as some monitoring processes may consist of a subnetwork of subprocesses.

1.1.2 Stable monitoring

The distributed processes may degenerate into chaotic interference without some means for structuring the interactions. To provide this structure, MAITA provides a "monitor of monitors" or "MOM" to construct, maintain, inspect, and modify the monitoring network and its operation. We achieve a degree of uniformity in the control process by organizing MOMs as special types of monitoring processes.

The MOM is designed to provide for resilient and persistent networks of monitoring processes. Toward this end, the command and control system monitors all the other monitoring processes, correcting and restarting them as needed. The control system itself is monitored by a subsidiary monitor which corrects and restarts the control system as needed. The architecture employs a persistent database to aid in providing this level of stability, and monitors the functioning of the database system as well. The control system also works to ensure the accuracy of the database records, both by updating them as changes are made and by checking them as needed.

1.1.3 Secure monitoring

The architecture provides for a fairly standard Unix-like scheme of users, groups, passwords, and permissions, specialized for the distinctive classes of operations performed on monitoring networks and their elements.

The architecture also is designed to provide a minimal target for would-be attackers by establishing most of its data communications through ephemeral listeners operating out of randomly-assigned ports. This leaves the only entry points of the system knowable in advance to be the main starting address of the system. Even this can be varied at the time of system startup, and across independent MAITA systems. Future improvements to the system may enable changing of this main address during system operation as well, allowing a form of "frequency hopping".

The initial design of the MAITA system presumes that the most security issues involving the control system must involve mechanisms external to MAITA that provide the operating context of the monitoring processes.

1.1.4 Open monitoring

The architecture is designed to provide an open platform for system development and interconnection. Command operations are transmitted using hypertext transport protocol (HTTP), allowing for basic system operation from any web browser, using commands entered by hand or through multiple specialized web pages or applets. Such web-based control minimizes requirements for installing specialized software on local machines. Supporting this open operation further, we provide reference implementations of the MAITA-specific communications mechanisms, in the form of Java and Common Lisp classes that provide monitoring process wrappers for use in legacy systems written in these languages.

Data are transmitted by an expandable set of common protocols, permitting direct interconnection with many legacy and separately-developed systems. The design permits information to flow through the network by several different protocols, including socket-based ASCII character streams, HTTP (Hypertext Transport Protocol, used by the World Wide Web), SMTP (the Simple Mail Transport Protocol, used by email systems), Java RMI (Java Remote Method Invocation), ODBC (Open Database Connectivity), and OKBC (Open Knowledge Base Connectivity, a protocol for transmitting logical and frame-structured knowledge to and from knowledge bases). The system developer or user chooses the protocol appropriate to the volume, regularity, and type of the information being transmitted. Regular and high-frequency transmissions typically go through persistent stream, ODBC, or OKBC connections. Intermittent and low-frequency transmissions probably go on temporary HTTP, SMTP, Java RMI, ODBC, or OKBC connections. Records of information transmitted to input or from output terminals are structured in protocol-dependent formats.

1.1.5 Intelligent monitoring

One can call any monitoring system knowledge-based, since its designers employ knowledge in the course of its construction. The MAITA architecture is intended to support additional roles for explicitly-represented knowledge, in the operation of monitoring systems.

The first role is that of monitoring processes which explicitly reason in the course of their analysis. MAITA supports these by offering an ontology of monitoring concepts and a knowledge base of monitoring methods. We annotate the structure of information flow with knowledge-level descriptors, distinguishing the *reports* being transmitted and received from the computational representations (*packets*) of these reports, and distinguishing these computational representations from the protocol-specific encodings used for transmission.

The second role is that of monitoring networks, in which the structure of the network explicitly reflects knowledge about the conditions being monitored. This network structure should identify the conditions of interest and the dependencies among them. For example, the structure of a monitoring network should revolve around the conditions on which attention should be focussed, and provide checking of expectations (both positive and negative) related to these conditions.

The third role is that of alerting models, in which knowledge about the likelihood of different classes of alerts or reports, time and other costs of transmission, and utility to different recipients or recipient classes is used to make rational choices about who to tell what, and when and how.

1.1.6 Evolutionary monitoring

Sensible engineering design calls for components that may be reused or adapted in subsequent designs. The MAITA libraries provide means for abstracting, recording, and sharing monitoring

networks developed for one purpose with developers of monitors for other purposes. These libraries aim to provide a broad and deep base of abstract and concrete monitoring methods, event descriptions, and alerting models.

Chapter 2

Monitoring concepts

This chapter sketches the fundamental monitoring concepts addressed by the MAITA technology and methodology.

2.1 Information flows

Monitoring systems involve several types of information flows: signals entering monitoring processes as inputs, alerts exiting monitoring processes as outputs, and information that processes send to and retrieve from local stores of memories.

2.1.1 Signals

We think of most information on which the monitoring system operates as type of signals. The types of data sources or signals of interest in monitoring tasks includes continuous signals, sampled continuous signals, text or message streams, propositional information, and graded propositional information, that is, with uncertainty (probability, evidence) or imprecision (fuzzy) measures. The signals of interest vary with the task, and most such tasks will involve only a subset of the possible types of data sources.

2.1.2 Alerts

Alerts represent the immediate results of monitoring, namely the signals sent to humans or other recipients to notify them of the occurrence of some event or the establishment of some condition. The MAITA architecture permits chaining of monitoring processes both sequentially and hierarchically, so that the results of one process can serve as a data source for other processes. Alerts can thus use more abstract languages than the initial input signals, though they may use exactly the same language when acting as pure filters on the input signals.

2.1.3 Local stores and memories

In addition to input signals and output alerts, monitoring processes may also use information in databases or knowledge bases that one interprets as background knowledge or memories instead of signal information. Processes may store statistics or sequences of inputs in memory to perform trend analysis or to facilitate explanations through roll-back and replay. Processes may also change background knowledge in the course of learning more about their environment and task.

2.2 Event recognition

Event recognition proceeds in stages, starting with simple transformations on individual signals to prepare them for more complex correlation and matching operations.

2.2.1 Transducers

Signal transducers transform a signal into one or more new signals. The most familiar variety of signal transducers all concern continuous or time-series signals. These include linear extrapolation and interpolation, trend-line fitting, wavelet decomposition, fourier transforms, summary statistics, outlier detection, threshold detection, and others.

The range of useful signal transducers appears to be more limited for propositional signals, including, for example, translation into new propositions, and measures of change statistics (for example, when last changed, how frequently changing). Sometimes propositions encode data that may be usefully viewed as a sampled signal; for example, sequential reports on the location of a person or piece of equipment may usefully be aggregated into a map-based data series and analyzed with corresponding techniques. One can view many low-level pattern-matching procedures into this category as well, including many internet firewall policies (for example, don't pass any incoming requests from `intruders.net`) and intrusion detection signature-recognizers and statistics collectors (for example, report all write attempts in `/usr/local`, port or IP sweeps, and syslog and ping of death attacks).

2.2.2 Correlators

one can think of signal correlators as multi-signal transducers that take several streams of data as inputs and provide one or more new signals (or propositions) as output. Correlators constitute one of the most important elements of a knowledge-based monitoring system. Events or trends of interest are normally realized in coordinated changes in different aspects of a situation. For example, common statistical abnormality detectors measure discrepancies between statistics at different time scales. As other examples, a discrepancy between rates of use and procurement of a resource can signal a problem needing attention, and determining a new possible motivation for an agent may cast the activity reflected in other data streams in a new light. Plan recognizers also are naturally viewed as looking for specific types of correlations between temporal events that characterize one or more execution paths through the plan. Correlators thus aggregate and abstract input reports to produce more informed output reports.

The building blocks of signal correlators include standard continuous-signal operations such as differencing, modulating, and demodulating, but the most interesting building blocks for knowledge-based applications are those correlating propositional and graded information, such as rules, reasons and argument structures, Bayesian probabilistic networks, causal networks, and temporal constraints.

2.2.3 Trend templates

We use these signal-correlating building blocks in a library of abstract and special signal correlators called *trend templates*, after the representation by that name developed at MIT by Haimowitz and Kohane in the `TrenDx` system [8, 7, 14, 11, 5]. A trend template (TT) is an archetypal pattern of data variation in a related collection of data that serves as a characterization of a type of event. For example, a particular information security trend template might characterize an event consisting of

a port sweep followed by increased traffic using some particular port to a small set of destinations rarely seen before.

Each TT has a temporal component and a value component. The temporal component includes landmark time points and intervals. Landmark points represent significant events in the lifetime of the monitored process. They may be uncertain in time, and so are represented with time ranges (min max) expressing the minimal and maximal times between them. Intervals represent periods of the process that are significant interpretation. Intervals consist of begin and end points whose times are declared either as offsets of the form (min max) from a landmark point, or offsets of the form (min max) from another interval's begin or end point. The temporal representation is supported by a temporal utility package (TUP) that propagates temporal bound inferences among related points and intervals [13, 12]. The value component characterizes constraints on individual data values and propositions and on computed trends in time-ordered data, and specifies constraints that must hold among different data streams.

2.2.4 Matching methods

In matching a trend template to data, two tasks are carried out simultaneously. First, the bounds on time intervals mentioned in the TT are refined so that the data best fits the TT. For example, a TT that looks for a linear rise in a numeric parameter followed by its holding steady while another parameter decays exponentially must find the (approximate) time boundary between these two conditions. Its best estimate will minimize deviations from the constraints. Second, an overall measure of the quality of fit is computed from the deviations. The measures of quality that tells how well various TTs fit the monitored data become either time-varying signal or propositional outputs of the signal correlators and trend detectors, and provide the appropriately processed inputs for making monitoring decisions.

The most appropriate language of trends and constraints varies from domain to domain. Our original constraint language included mainly linear and quadratic regression models for numeric data, absolute and relative numerical constraints on functions of the data, and logical combinations of such descriptions and propositions. Our newer additions develop the ability to build other TTs using descriptions that characterize any outputs of signal transducers and additional models of correlation among signals. We intend that the template library will grow over time, with research and new applications leading to new additions. Augmenting the library with new templates forms one of the key operations in using the system to address special and temporary concerns.

2.2.5 Recognition control

While one might construct the simplest sorts of monitors to perform operations on input signals in a fixed fashion, event recognition in more complicated situations requires that the recognition operations vary with changing circumstances, that is, that the recognition process possesses a degree of situational awareness and exhibits a degree of situation dependence in its operation. In fact, we can view almost every multi-signal correlation process as exhibiting situational awareness by interpreting some of its inputs in the context of the "situation" represented by the other inputs.

We claim that effective monitoring in many real-world domains requires that at least some of the monitoring processes exhibit situational awareness in a broader sense, in which changes in monitor behavior are triggered by sporadic receipt of updates about a range of relevant conditions occurring in the environment of the monitor. Such updates may take the form of changes to background information rather than receipt of explicitly directed signals. In the information security arena, such updates might be as simple as a change of "infocon" levels akin to "defcon" levels, or as

complex as propositional or probabilistic updates to a situational knowledge base maintained by the monitoring process. We can of course view the sequence of such updates as just another input to the monitor, but the sporadic and nonuniform nature and discontinuous effects of such updates, in which they change the way future inputs are processed, make it more natural to view the updates as changing the situational model employed by the monitor in processing the ordinary input signals.

We distinguish several forms of situational awareness useful in monitoring processes. The first dimension of variation divides monitors according to whether the situation in question is an "objective" or "intentional" situation. In the information security domain, the objective situation might consist of information about whether related enclaves are experiencing attacks or whether internet congestion levels seem abnormally high; the intentional situation might consist of the preferences of a security officer regarding the seriousness of abnormalities required to justify issuing an alert on the security desk.

The second dimension of variation divides monitoring methods according to whether the situational model maintained by the process consists of only summary variables or a model of some complexity. In the information security domain, summary variables might include overall probability and disutility of an attack at present, source and target of attack, timing, purpose, and method of attack, and level of defenses available. More complex models might characterize threats using an influence diagram that expresses the overall probability and utility of attack in terms of information about attacks on other enclaves, news articles about increased tensions with known adversaries, and social and infrastructure disruptions such as power outages and strikes.

Degree of passivity forms another important dimension of variation. At one extreme, simple monitors based on lists of indicators and warnings may just observe a set of propositional inputs to detect the presence or absence of a set of specific conditions, and the output of the procedure is to simply report the set of present conditions, or perhaps just the number of conditions present at a given time. At the other extreme, active monitors may start with such a list of indicating conditions and continuously actively seek out new information to determine the presence or absence of these conditions, as opposed to simply waiting for notifications of presence to enter as inputs. Intermediate monitoring procedures might simply filter inputs passively until some threshold is reached, and then switch to an active mode to confirm or deny the remaining conditions.

Degree of passivity is closely tied to notions of the utility of information. Once an active search is underway, the best strategy is to seek first the information most useful to answering the question in the time allowed, but utility considerations also arise in formulating the thresholds at which monitors "go active". For example, with only a few pieces of information, learning an additional item on an indicators list may not change the quality of the match significantly. But at some point, learning an additional item makes each of the remaining items very significant, and "going active" at that point may well be the appropriate path.

2.3 Event reporting

Useful event reporting relies on informed decisions about what events merit reporting and on perspicuous reporting media.

2.3.1 Alerting control

Recognition of an event calls for deciding what to do with that information: who to notify, when to notify them, and how to notify them. Tailoring the methods used for making alerting decisions constitutes a key method for making the monitoring system responsive to individual analysts.

Most of the effort in guiding alerting decisions consists in describing the utility of different results to different agents in different situations.

Alerting decisions depend on the event being reported since analysts have priorities among the conditions of interest to them, and normally wish to hear about the most urgent and important items right away, with the lesser items deferred for consideration later. Alerting decisions depend on the recipient since different analysts will have different interests, priorities, and tasks. Alerting decisions may also depend on the sets of possible recipients and media used to communicate alerts. For example, unreliable transmission or receipt times may call for copying alerts to backup recipients or through alternative transmission media.

2.3.2 Displays

Human analysts require the results of analyses to be presented in intelligible forms, such as graphs and charts that convey the important information prominently without dilution by extraneous detail. The MAITA system presently employs several main display types following common forms, but construction of optimized displays has not been a focus of our effort.

MAITA provides *multivariate strip charts* of selected streams that display the values of one or more variables on a rectangular graph, with the displayed variables plotted vertically and time plotted horizontally. The system can operate individual strips as well as combination strips in which several individual strip charts are stacked one on top of the other with a common temporal reference on the horizontal. The system provides the ability to create combination and multivariate strip charts by selecting various connections or terminals in the process network diagram and then performing the appropriate control-menu operation.

MAITA provides *two-dimensional (2D) maps* of variables plotted against each other that display one or more paired variables, with one set of variables plotted on the vertical, and another set plotted on the horizontal. In a 2D map, time does not appear as a dimension of the graph axes. Instead, the temporal window appears only through the number of points plotted; as the temporal window moves, excessively old points are removed from the display, and the new points are added.

Text alerts display sentences, phrases, or words that constitute alerts to the user.

MAITA also supports combinations of these types. For example, one might combine a strip chart with a text alert that states the normality or abnormality of the stream contents being displayed in the strip chart.

We expect to make future extensions that provide additional visual display types, as well as nonverbal audio and synthesized speech alerts.

2.4 Monitoring knowledge

Different but related bodies of knowledge may be involved in constructing and operating monitoring systems. The monitoring processes themselves may make use of situational knowledge bases for sharing or reasoning about their situations. To aid human developers or analysts in constructing, maintaining, operating, or tailoring monitoring systems, this situational knowledge is less important than knowledge about the varieties of possible monitoring, alerting, and display methods, including descriptions of standard types of events of common interest and means for reporting these events.

2.4.1 Libraries

Libraries of event descriptions, recognition methods, alerting models, and display models constitute the core of the monitoring knowledge of concern to developers constructing new monitoring systems

and to analysts tailoring monitoring systems to their own needs.

Event descriptions

There is little to say about the event description library in the abstract since almost everything it contains represents an element of knowledge specific to some domain. The top levels of such event descriptions correspond to the main sorts of events found in everyday language, such as starts, stops, attacks, defenses, infections, cures, weakenings, strengthenings, victories, and defeats. Each body of domain-specific monitoring information adds specific subtypes and instances of such events. Our focus here lies in constructing trend templates to describe interesting and useful event classes, but the library may contain other types of event descriptions as well, such as Bayesian networks that provide probabilistic characterizations of events.

Recognition methods

The recognition method library includes entries at all levels of computational detail, from very abstract procedures covering virtually all monitoring tasks, to intermediate-detail procedures capturing more specific algorithmic ideas and domain-specific information about the types of signals being monitored, to highly detailed procedures involving specific representations, code, domain details, and signal sources. For example, the most abstract levels might speak of constructing and comparing a set of hypotheses about what is going on, without providing any details about how the hypotheses are constructed or compared. At an intermediate level, the Trend_x [8, 7, 14, 11, 5] trend monitoring system developed at MIT uses a partial-match strategy operating over a set of trend templates, each of which consists primarily of temporal constraints characterizing some temporal event. More refined monitoring models would emend this procedure to take probabilistic or default information into account, or to embed background knowledge of the domain in the matching strategy (for example, always try matching location information first before bothering with other information). Still more concrete procedures might describe operating-system-specific methods for recognizing port sweeps or ftp-write attacks. The most abstract control and interpretation procedures serve as a base for more specific ones, but one rarely uses them directly. The real strength of the library of monitoring models lies in identifying specific combinations of representations, procedures, and domain characteristics that offer significant power compared to the more abstract procedures. We expect a fully developed library to contain a great many entries.

Procedures for a small number of fairly abstract monitoring procedures have been codified already in the CommonKADS library of problem solving methods [1], but most of these concern fairly active procedures for diagnosing devices for which complete structural and functional information is available. Such complete information does not exist for some important medical and information security applications.

Combining and cascading monitoring procedures leads to additional library elements, since one may sometimes combine synergistic but separate monitoring procedures into more effective ones. Some of these combinations on monitoring procedures mirror combinations of trend templates, thus reflecting portions of the event description library, but the dataflow connections among monitoring procedures and algorithmic variation in the methods mean that the monitoring procedure library stands on its own rather than as a derivative of the event description library.

Library entries include two types of descriptions of monitoring methods: competence or capability descriptions, and performance descriptions. The capability descriptions characterize primarily the types of information on which the method operates, the relations of inputs to outputs, and the purposes or principal uses of the method. The performance descriptions, in contrast, character-

ize the representations or data structures used to encode information and the procedural steps or concrete code used to execute the method.

Alerting models

The library of alerting models incorporates both extant procedures for making alerting decisions and methods for convenient specification of utility information. The medical informatics literature contains an unsystematic variety of alerting procedures, with few tied to explicit notions of utility (see, for example, [10, 9, 15, 16, 17]). Our ongoing construction of alerting models uses explicit utility models to develop a systematic collection of alerting procedures that includes the ones already reported in the literature. The representations here build on our past work [18, 3, 19, 4] on qualitative representation of utility information, which has developed logical languages that can express generic preferences ("prefer air campaign plans that maintain a center of gravity over those that distribute forces more widely"), and that relate this notion of preference to the notion of problem-solving or planning goals (interpreting goals as conditions preferred to their opposites, other things being equal). We are developing utility models that combine both qualitative preference information with approximate numerical models of common utility structures (for example, utility models that increase up to some time and then drop off to model deadline goals, as in [6]), along with automatic procedures for combining such information into qualitative decision procedures and numerical multiattribute utility functions suitable for quick evaluation of alternatives.

2.4.2 Ontology

If the libraries of event, recognition, and alerting models form the core of systemic monitoring knowledge, the monitoring ontology forms the backbone of both these libraries and knowledge bases that represent the current monitoring situation.

To formalize the monitoring libraries we employ an ontology for monitoring processes, including concepts such as causal structures (chains constitute only the simplest such structures), partial matches, evaluations of significance and likelihood, and focus of attention. This requires a rich language in which to express monitoring and alerting procedures, a broad body of world knowledge with which to relate different monitoring and organizational concepts, and a clear organization of the procedures themselves that reflects abstraction hierarchies and other dimensions along which to classify the procedures. Reference to a body of concepts about the world is essential for specifying the intent of different monitoring procedures, and for increasing the coherence of the library. The organization of the monitoring procedures according to different properties and dimensions of variation is essential for reasoning about and manipulating these descriptions in the course of knowledge acquisition, learning, compilation, and other aspects of the process of maintaining a monitoring system. Similar remarks apply to models of organizational structure and function, which monitors may employ in expressing security policies and judging appropriateness of communications and other operations.

Representing complete domain-specific monitoring situations formally requires even more ontological development, but most of that lies outside the scope of the MAITA project.

2.4.3 Editing tools

To permit easy augmentation and refinement of the set of monitors and bodies of monitoring knowledge and procedures, the MAITA system provides a set of editing tools for creating, copying, removing, filling out, and revising trend templates, monitoring models, and alerting models, as

well as a set of informational tools for querying existing ontologies, knowledge bases, and reference materials.

Constructing trend templates involving propositional conditions requires a system for representing these conditions. For practical use, this means having on tap one or more formalized knowledge bases and ontologies to provide the vocabulary and background information needed to express the monitoring conditions. The MAITA system builds on existing tools for editing knowledge bases, augmenting these tools with tailored interfaces for specific types of descriptions, such as event, recognition, and alerting models.

Chapter 3

Demonstration

This chapter describes a demonstration of the MAITA system functionality prepared in September 1999.

The functionality described here superseded that of an earlier prototype design that employed web-backed relational database scripts as the primary medium of computation. We presented results from an initial web-database battlefield movement analysis system at the 6-month meeting. This demonstration provided some of the basic tools for our later work, though it also showed the database was too slow for the intended application. Accordingly, we changed our architectural plans to use the database only for persistent storage of the control state of the system (though not precluding individual monitoring processes from using databases in other ways).

This document exhibits some of the functionality of the MAITA implementation and graphical user interface at that time through annotated snapshots of an actual demonstration. The primary demonstration provided here illustrates the creation and running of a monitor for detecting pneumothorax (collapsed lung) in infants. An additional demonstration, not illustrated here, provides a depiction of the movement of convoys on a network of roads. Results of that system are reported in [2].

The MAITA system employs a persistent server called the MOM to manage the operations of distributed monitoring processes. The demonstration overview below presumes the MOM is already operating. One typically exerts control over monitoring operations by means of the MOM control panel, which consists of a Java applet running in a web browser. Both the MOM server and control panel applet are works in progress. The appearance of the control panel is especially likely to change from that presented in the following.

The pneumothorax monitor consists of three major components subdivided into seven processes.

- The first component consists of the two data sources, one process simulating a blood-gas probe device measuring blood oxygen and carbon-dioxide levels, and a second process simulating a cardiac-activity probe measuring heart rate and mean blood pressure.
- The second component consists of the monitoring processes proper, namely an artifact detector (noise eliminator) feeding a pneumothorax detector. These two processes form subprocesses of a compound superprocess representing the filtered pneumothorax detection operation.
- The third and final component of the pneumothorax monitor consists of two predefined display processes, one for displaying the alerts generated by the pneumothorax detection process, and another for displaying the signals from the heart-activity probe. These predefined displays do

not exhaust the possible displays, for one can also display the signals coursing through any terminal of the network of monitoring processes.

To summarize the presentation, we show how to log onto the MOM to obtain a MOM control panel; how to browse the library of monitoring processes to create and run an instance of the pneumothorax monitor; how to display the signals traversing the monitoring network; how to control the individual processes; and how to quit monitoring.

3.1 Logging in

Opening the MOM control panel in a web browser displays the login panel shown in Figure 3.1. One types a valid login name and password in the indicated areas and then clicks the Enter button.

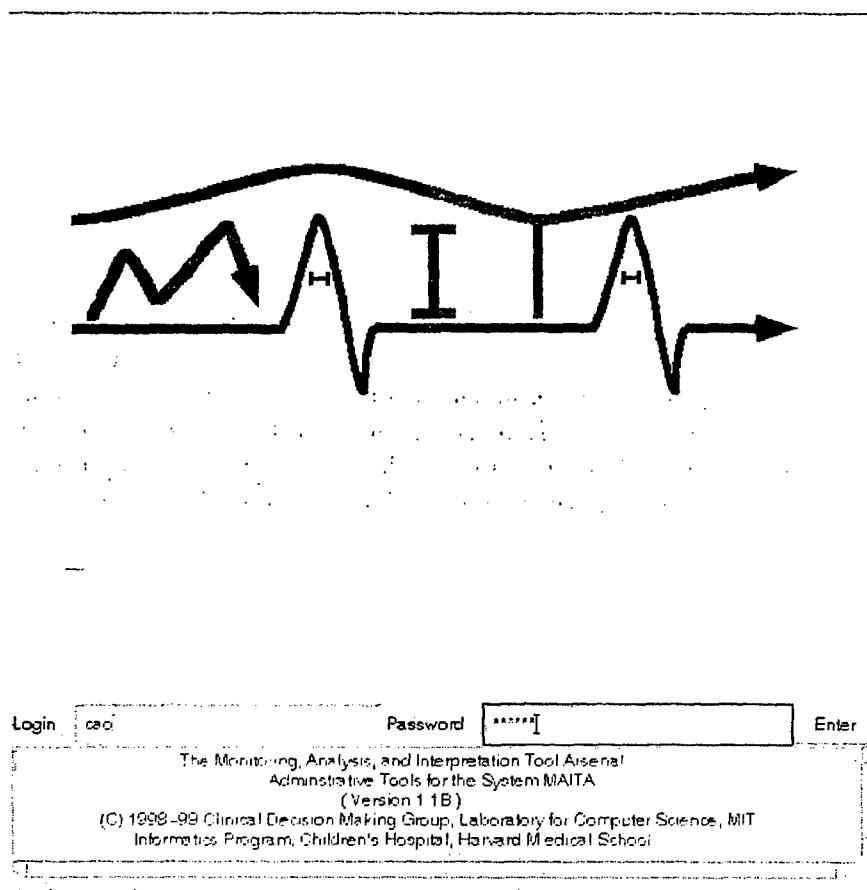


Figure 3.1: MOM Login Panel

3.2 Creating a monitoring network

The MOM control panel then displays the monitoring network editing panel depicted in Figure 3.2. The main network-display panel appears blank if there are no monitoring processes currently running, displays the running processes if any exist.

The next step is to locate some process descriptions in the library to create monitor instances to run. To do this, click on the Edit button.

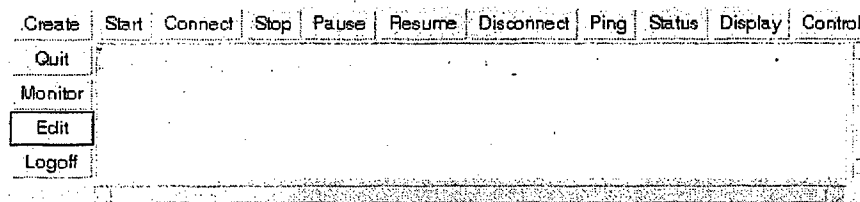


Figure 3.2: Network Display Panel

3.3 Locating a library process

The MOM control panel now displays the process library browser panel shown in Figure 3.3. Double-clicking the Pneumothorax-Monitor item selects that process description from the library. At this point, one can edit the entries in the text areas of the display to change the characteristics of the selected description. One now closes the library browser panel by clicking the Quit button on that panel.

3.4 Examining the new network

The network panel now displays the pneumothorax monitoring process network as shown in Figure 3.4. Here gas-probe and cardio-probe denote the signal sources, af-detector denotes the artifact detector, pt-detector denotes the pneumothorax detector, and ptd-1 and ptd-2 denote the predefined display processes.

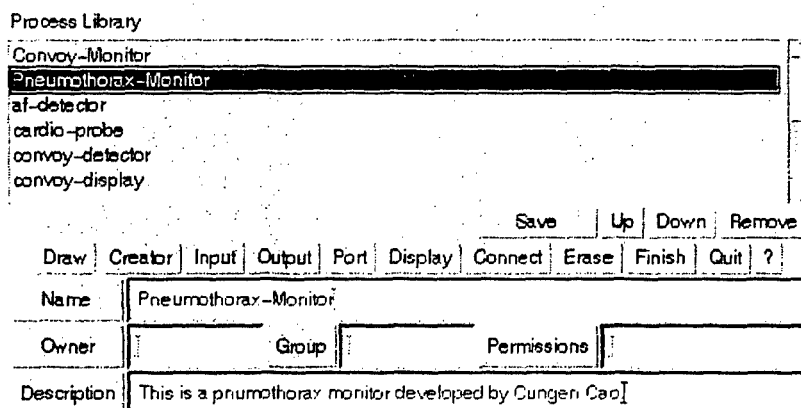


Figure 3.3: Library Browser Panel

In this case, the network depiction fits within the network display window. If the network depiction is too large to fit, the MOM control panel displays only a portion of the process network, and one must use the network navigator panel described later to change the area under examination. One accesses the network navigator panel by clicking the Monitor button:

3.5 Navigating the network display

Figure 3.5 shows the network navigation panel. Clicking on any position on the navigation panel focuses the network depiction on a portion of the process network centered at that point. One may use the Close toolbar item to close the network navigator display.

3.6 Starting the monitor

In the current MOM control panel, instantiating the library description to form a network of monitoring processes just creates icons in the network display, and does not actually create the monitoring processes or start them running.

To create actual processes corresponding to the network display, one first selects the processes to create using the panel given in Figure 3.4. In this case, clicking on the outermost monitor name Pneumothorax-Monitor selects all the processes involved in the pneumothorax monitor. One then clicks on the Create button and waits until all the process icons turn green, which signals they have been created. The creation step can take a minute or two in the present implementation. To set the new processes in motion, one selects the processes to run (already done in this case) and clicks on the Start button.

Pneumothorax-Monitor

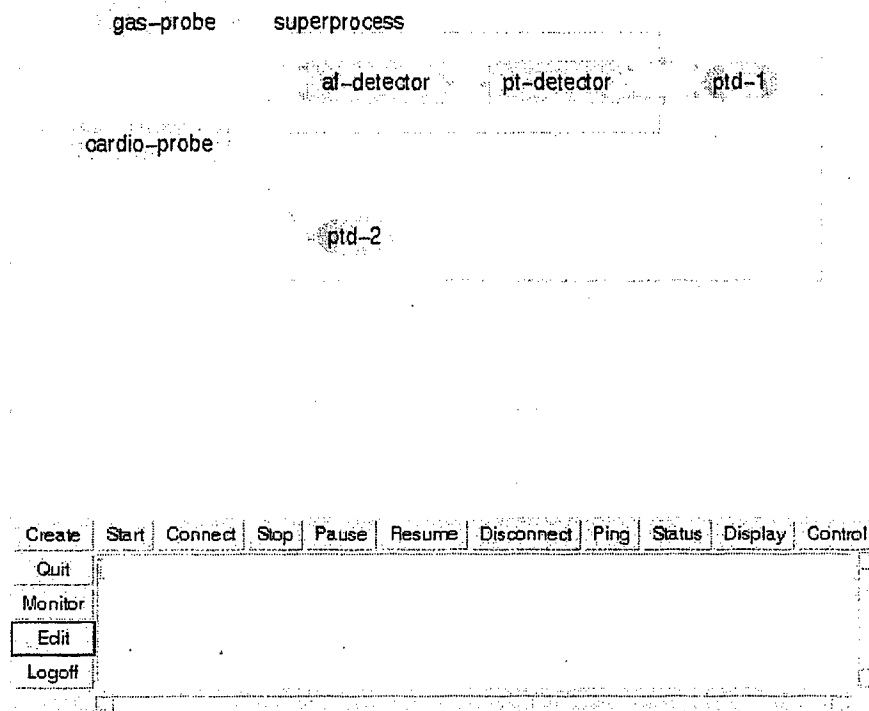


Figure 3.4: Pneumothorax Monitoring Process Network

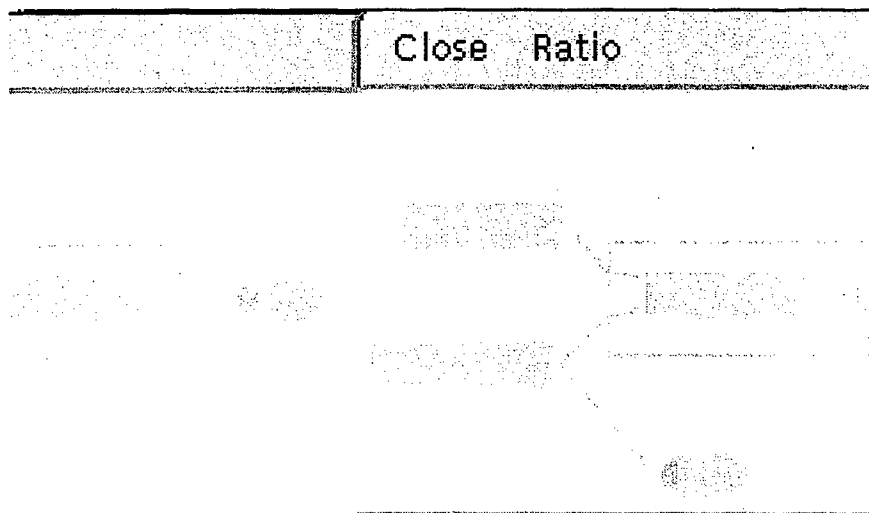


Figure 3.5: Network Navigation Window

3.7 Displaying information flows

The MOM provides means for displaying signals flowing through the monitoring network. Each monitor type in the library defines default display types for each of the signals received and produced. One can define specialized or variant displays to augment or override these in specific applications.

The pneumothorax monitor provides a default display for the gas probe terminal and specialized displays for the cardiac probe signals and the pneumothorax detector outputs.

To examine the signal on the gas probe terminal, one selects that terminal by clicking on the square terminal mark to the right of the gas probe icon, after first making sure nothing else is selected by clicking on any selected items to deselect them. With this terminal selected, click on the Display button to create the display window, as indicated in Figure 3.4.

3.8 Strip charts and text alerts

Click on ptd-1, and then click on the Display button. You will see a display similar to Figure 3.6 providing strip charts for the carbon dioxide and oxygen levels in the patient's blood stream, together with a temporal text window announcing the occurrence of pneumothorax events.

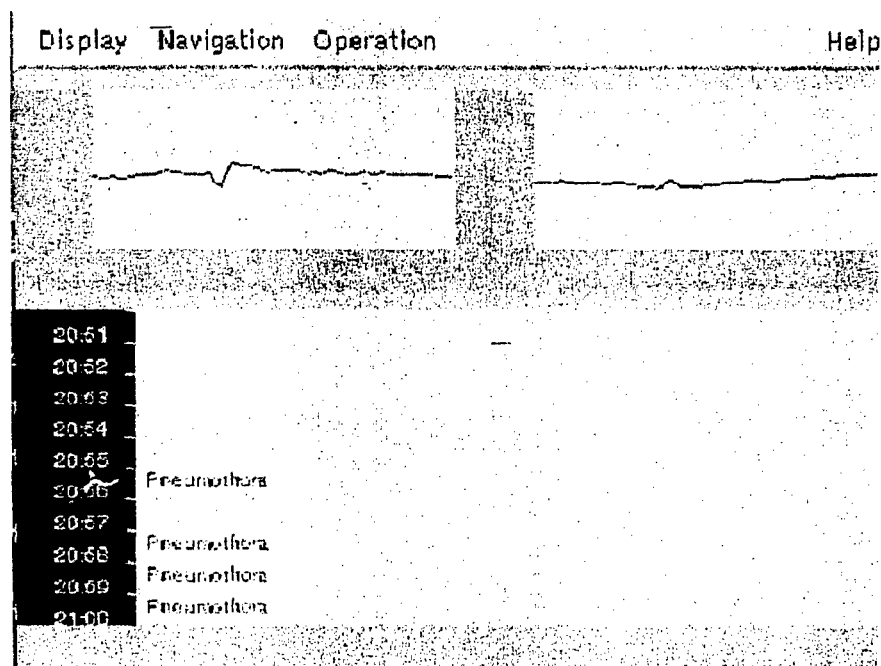


Figure 3.6: Displaying terminal signals and pneumothorax alerts

3.9 Two-dimensional plots

To see a two-dimensional plot of one variable against another, with the number of displayed points corresponding to the width of the temporal window, select ptd-2 and then click on Display. This produces a display similar to the one shown in Figure 3.7. Two-dimensional plots may be made

against a background depicting areas of interest, such as safe or dangerous control regions in patient care, or geographic regions and features. Figure 3.8 shows a road network from a battlefield movement monitoring application.

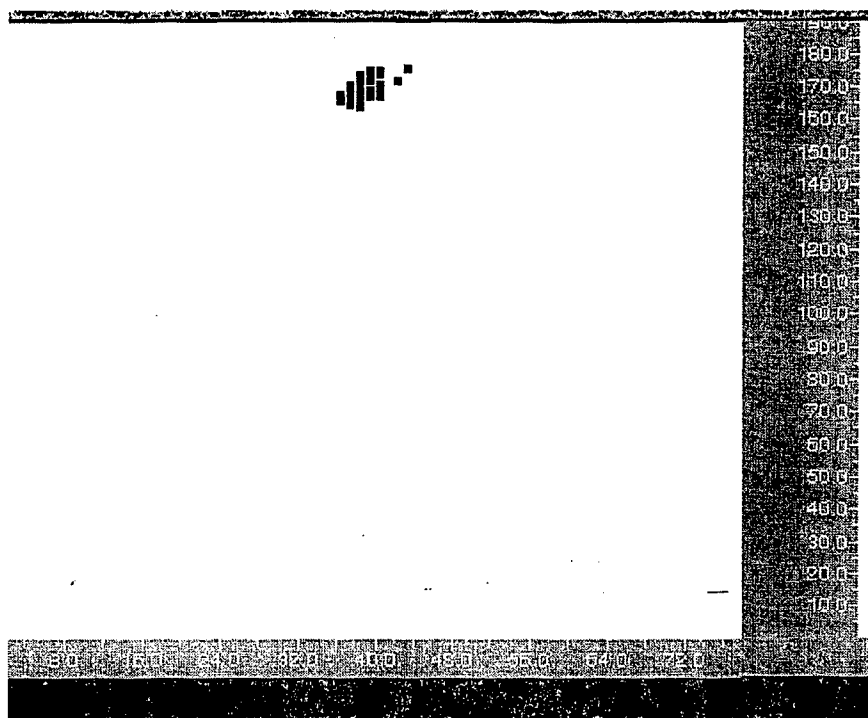


Figure 3.7: Heart rate vs. Mean Blood Pressure

3.10 Controlling processing

The MOM provides a number of ways to control and inspect the operation of monitoring processes. Ping checks to see if the process is still functioning. Status provides a brief summary of the operational status of the process. Pause temporarily halts processing by the selected process or terminal, and Resume resumes it. Start sets a process in motion, and Stop has it cease operations. Connect creates a data connection between two selected terminals. Disconnect sunders selected connections.

3.11 Quitting processes and logging off

To quit processing, select all the icons of process instances, and then click the Quit button. This will destroy the processes and remove them from the MOM's database. One then logs off from the MOM control panel by clicking the Logoff button. This returns one to the initial login screen.

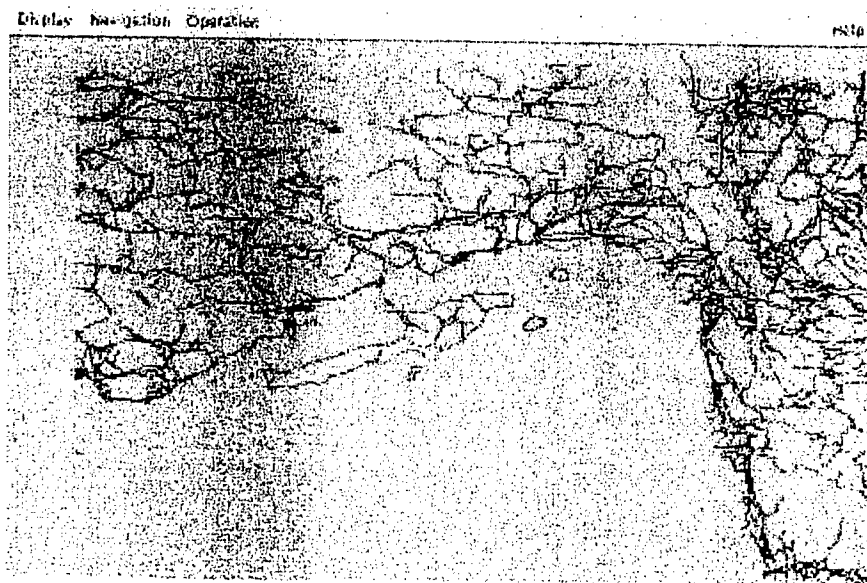


Figure 3.8: Geographic Background Display

Chapter 4

Applications

The MAITA system was developed in conjunction with two types of monitoring application, first battlefield movement analysis, and then intrusion detection and information assurance. Work on these applications also brought us into contact with other applications, and we attended a Project Genoa briefing to learn about monitoring issues in intelligence analysis and crisis management.

4.1 Battlefield movement analysis

The MAITA system was initially used to construct systems aimed at monitoring battlefield movements in order to alert commanders to high-value events such as major troop movements and displacements of antiaircraft batteries, mobile missile launchers, and mobile radar units. Initial versions of the monitors so constructed were tested on simulated data.

We participated in meetings with Army subject-matter experts to learn about movement analysis tasks and methods. We learned about the nature of the recognition tasks, and about what the experts believed to be important. Part of this was a classification of the pattern recognition tasks according to military interest. The top categories were any delivery mechanisms for WMD and the ADA sites protecting those mechanisms; the second rank of categories were command posts and reserve mech and armored divisions; and everything else was in the noise except as clues to identifying these higher priority targets. We learned what tasks are hard for human analysts, and why. Humans can see big, long lasting movements easily enough themselves, but they can't see the small, irregular movements (such as reloading and relocating a Scud launcher) lost in the sea of motion, hidden by a flood of data and by division of attention among different analysts and across different shifts. Since many of the clues to the high priority sites were exactly of this type that is hard for people but may be easy for a computer with a memory tailored to looking for these events, we came away with the belief that any progress on these tasks would be viewed as a great accomplishment by consumers of this technology. If we consider how poor our Scud locating efforts were supposed to have been in Desert Storm, it seems that hopes of doing considerably better would have to raise a lot of interest.

We also learned that military analysts would evaluate system performance in terms other than accuracy identifying all movement patterns (regardless of type) and speed of the computational process. The analysts said the evaluation criteria depend on the recipient of the notifications. Intelligence officers want to hear everything as quickly as possible, even if very uncertain or abstract. They then apply their much deeper knowledge to filter out these reports and build up a coherent picture. In contrast, commanders don't want this; they want the intelligence officer's best estimate of the situation, or a very small number of alternatives. With this advice on evaluation, we aimed

to support the intelligence officers. They are the ones suffering the data flood, and they have the knowledge and the responsibility for putting the overall picture together. Since we are unlikely in the near term to have the broad and deep knowledge needed to supplant intelligence officers in preparing estimates for commanders, the intelligence officer evaluation model seems the right one to aim for.

We also investigated techniques in computational geometry as means for locating movement signals with respect to a road network, and supplied Java and Lisp routines that several HPKB groups used for this purpose.

We demonstrated early prototypes of the MAITA architecture and tools in June 1998 in a system for multi-sensor information fusion of simulated battlefield information. That system took in streams of moving-target indicator (MTI) data from a JSTARS unit, interpreted these signals using knowledge about terrain, order of battle, fixed installations, and correlated the signals with intelligence reports derived from humans, SIGINT, ELINT, and photogrammetric sources to identify military sites, routes, vehicles, site types, displacements of specialized units, etc.

We used information gained from this demonstration to improve on the underlying architecture, and developed an implementation of the revised architecture and network control tools. We also refined the language for describing monitoring methods and expanded the library design to include a broader range of methods.

The improved monitoring architecture and tools, described earlier in Chapter 3, were then used in a challenge problem evaluation conducted in 1999. Our work provided key geographical computation routines used by several HPKB participants in the challenge problem evaluation. The movement analysis methods implemented with the MAITA system proved of comparable effectiveness to those developed by other participants, though ambiguities in the specification of the reporting format led to variations in the accuracy of evaluation across the groups. Some of the methods implemented in MAITA performed as well as any; some specialized methods failed to perform correctly for reasons never determined. While the evaluation allowed demonstration of some MAITA capabilities, the evaluation itself did not really assess the properties guiding the design of the MAITA system tools as opposed to specialized programs specifically designed for geographic computations.

4.2 Information assurance

With the replacement of the HPKB battlefield movement analysis challenge problem by a problem not involving temporal monitoring, we were encouraged to look at the information assurance domain as an arena in which to develop the MAITA monitoring technology.

We made early contributions to the HPKB program by reporting on possible application of HPKB technologies in the IA domain. One consequence of this analysis was retargeting the MAITA project to work in the IA domain, in which temporal monitoring for intrusion detection and attack correlation play large roles.

We also made early contributions to the IA field by examining the practice of intrusion detection systems, formalized in the Common Intrusion Specification Language (CISL), of reporting numerical grades of certainty associated with intrusion reports. Experience with diagnostic reporting in other application areas showed that such certainty reports have little value unless the reporting system distinguishes the notions of likelihood ratios of accuracy from posterior probabilities. We continued this work with effort aimed at gaining more knowledge of IA tasks, and consulted with an active IA groups at Lincoln Laboratory, SRI, and UCSB to learn more about the area.

We obtained data sets from the MIT Lincoln Laboratory Intrusion Detection System Evaluation

effort. We used these data sets to develop and test initial monitoring methods for gathering and exploiting trend information in the IA domain. During this period, we also used a small medical monitoring application using real ICU data to test the developing architecture and methods. We reported on this work at the final HPKB PI meeting, and our efforts in the area were ongoing at the conclusion of the HPKB effort. We also reported on this work and domain at the MIT LCS annual meeting.

Our HPKB/IA work in this year has three specific aims: further completing and improving the MAITA system, reconstructing existing IA monitoring systems in the MAITA framework, and testing utility for recognition of data misuse.

Our work on reconstructing existing IA monitoring systems in the MAITA framework involved reproducing some of the behavior of existing intrusion detection systems by implementing and running a set of detectors over portions of the Lincoln Labs database test sets. We developed some of these event detectors ourselves, but also tried to obtain formalized descriptions from existing IA efforts.

We also made initial investigations aimed at testing the utility of MAITA methods in recognizing data misuse. This involved codifying models of an organization so as to help detect inappropriate activity by agents (persons or programs) within it. The main focus of that investigation was modeling the structure of Childrens Hospital, for which we had access to sets of de-identified database access logs suitable for examining to find inappropriate accesses. While the database accesses in this set were all by humans (to the best of our knowledge), we sought to characterize the standard range of appropriate accesses in terms of a model of the organizational and functional structure of Childrens Hospital. We successfully captured several important rules for recognizing misuse, but abandoned the data misuse task in order to concentrate on intrusion detection and correlation tasks of greater interest to the Darpa IA effort.

4.3 Additional developments

In addition to using movement analysis and intrusion detection data sets for developing and testing monitoring methods, we also made use of medical data sets readily available to us through our contacts in Boston's Childrens Hospital. The primary data sets used consisted of neonatal intensive care unit (NICU) monitoring signals, pediatric intensive care unit monitoring signals, and pediatric growth data.

The NICU data was used by Cungen Cao and Neil McIntosh to develop methods for detecting lung collapse, and provided the main set of test data for developing the MAITA strip-chart and state-space displays.

The pediatric ICU data was used by Christine Tsien in her doctoral work which showed the value of human guidance of the data-mining process. Standard methods for inducing decision trees for recognizing events in data often produce rule sets humans find unintelligible. Tsien's work showed that by involving a knowledgeable human in the induction process, one can find decision trees of comparable performance that use concepts intelligible to humans.

The pediatric growth data was used by Mary De Souza in her masters thesis to improve the Trendx trend recognition system developed earlier by Haimowitz. Further improvements of the Trendx code have served as important components of the MAITA system.

Chapter 5

Conclusion

The MAITA project seeks to develop a fairly comprehensive suite of tools for constructing, operating, and modifying distributed monitoring systems. Our work in the HPKB effort made progress on both developing the general architecture and implementation and on developing applications to monitoring battlefield movements and intrusions in computer systems.

Bibliography

- [1] J. A. Breuker and W. Van de Velde, editors. *The CommonKADS Library for Expertise Modelling*. IOS Press, Amsterdam, 1994.
- [2] P. Cohen, R. Schrag, E. Jones, A. Pease, A. Lin, B. Starr, D. Gunning, and M. Burke. The darpa high-performance knowledge bases project. *AI Magazine*, 19(4):25-49, Winter 1998.
- [3] J. Doyle, Y. Shoham, and M. P. Wellman. A logic of relative desire (preliminary report). In Z. W. Ras and M. Zemankova, editors, *Methodologies for Intelligent Systems, 6*, volume 542 of *Lecture Notes in Artificial Intelligence*, pages 16-31, Berlin, Oct. 1991. Springer-Verlag.
- [4] J. Doyle and M. P. Wellman. Representing preferences as *ceteris paribus* comparatives. In S. Hanks, S. Russell, and M. P. Wellman, editors, *Proceedings of the AAAI Spring Symposium on Decision-Theoretic Planning*, 1994.
- [5] J. Fackler, I. J. Haimowitz, and I. S. Kohane. Knowledge-based data display using trendx. In *AAAI Spring Symposium: Interpreting Clinical Data*, Palo Alto, 1994. AAAI Press.
- [6] P. Haddawy and S. Hanks. Representations for decision-theoretic planning: Utility functions for deadline goals. In B. Nebel, C. Rich, and W. Swartout, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 71-82, San Mateo, CA, 1992. Morgan Kaufmann.
- [7] I. J. Haimowitz and I. S. Kohane. Automated trend detection with alternate temporal hypotheses. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, pages 146-151, Chambéry, France, 1993.
- [8] I. J. Haimowitz and I. S. Kohane. An epistemology for clinically significant trends. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 176-181, Washington, DC, 1993.
- [9] B. Hayes-Roth, S. Uckun, J. E. Larsson, J. Drakopoulos, D. Gaba, J. Barr, and J. Chien. Guardian: An experimental system for intelligent ICU monitoring. In *Symposium on Computer Applications in Medical Care*, Washington, DC, 1994.
- [10] B. Hayes-Roth, R. Washington, D. Ash, R. Hewett, A. Collinot, A. Vina, and A. Seiver. Guardian: A prototype intelligent agent for intensive-care monitoring. *Journal of AI in Medicine*, 4:165-185, 1992.
- [11] I. Kohane and I. Haimowitz. Hypothesis-driven data abstraction. In *Symposium on Computer Applications in Medical Care*, Washington, DC, 1993.

- [12] I. S. Kohane. Temporal reasoning in medical expert systems. In R. Salamon, B. Blum, and M. Jørgensen, editors, *MEDINFO 86: Proceedings of the Fifth Conference on Medical Informatics*, pages 170-174, Washington, Oct. 1986. North-Holland.
- [13] I. S. Kohane. Temporal reasoning in medical expert systems. TR 389, Massachusetts Institute of Technology, Laboratory for Computer Science, 545 Technology Square, Cambridge, MA, 02139, Apr. 1987.
- [14] I. S. Kohane and I. J. Haimowitz. Encoding patterns of growth to automate detection and diagnosis of abnormal growth patterns. *Pediatric Research*, 33:119A, 1993.
- [15] S. M. Ornstein, D. R. Garr, R. G. Jenkins, P. F. Rust, and A. Arnon. Computer-generated physician and patient reminders. *Journal of Family Practice*, 32:82-90, 1991.
- [16] D. M. Rind, C. Safran, R. S. Phillips, W. V. Slack, D. R. Calkins, T. L. Delbanco, and H. L. Bleich. The effect of computer-based reminders on the management of hospitalized patients with worsening renal function. In P. Claytons, editor, *Proceedings Symposium Computer Applications in Medical Care*, pages 28-32, Washington, DC, 1991. McGraw-Hill.
- [17] D. M. Rind, C. Safran, R. S. Phillips, Q. Wang, D. R. Calkins, T. L. Delbanco, H. L. Bleich, and W. V. Slack. Effect of computer-based alerts on the treatment and outcomes of hospitalized patients. *Archives of Internal Medicine*, 154:1511-1517, 1994.
- [18] M. P. Wellman and J. Doyle. Preferential semantics for goals. In *Proceedings of the National Conference on Artificial Intelligence*, pages 698-703, 1991.
- [19] M. P. Wellman and J. Doyle. Modular utility representation for decision-theoretic planning. In *Proceedings of the First International Conference on AI Planning Systems*, 1992.

Appendix A

List of personnel

The following people were supported by contract F30602-97-1-0193.

1. Jon Doyle, principal investigator
2. Peter Szolovits, co-principal investigator
3. William J. Long, principal research scientist
4. Isaac Kohane, research affiliate
5. Cungen Cao, research affiliate
6. Philip Greenspun, graduate student
7. Andrew Nakrin, graduate student
8. Christine Tsien, graduate student
9. Mary T. DeSouza, undergraduate student
10. Ying Zhang, undergraduate student
11. Heather Grove, secretary

Appendix B

List of contract publications

The following publications were produced either with support from contract F30602-97-1-0193 or in initiating the project.

1. Jon Doyle, Isaac Kohane, William Long, and Peter Szolovits, High-Performance Knowledge Base Support for Monitoring, Analysis, and Interpretation Tasks, December 4, 1996.
2. Jon Doyle, Isaac Kohane, William Long, and Peter Szolovits, The Architecture of MAITA: A Tool For Monitoring, Analysis, and Interpretation, draft of September 21, 1999.
3. Jon Doyle, Isaac Kohane, William J. Long, and Peter Szolovits, Adaptive Knowledge-Based Monitoring for Information Assurance, October 30, 1998.
4. Peter Szolovits, Detectors should be characterized by likelihood ratios, not posterior probabilities, June 18, 1999, revised February 1, 2000.
5. Jon Doyle, Some Representational Limitations of the Common Intrusion Specification Language, October 26, 1999, revised November 5, 1999.
6. Jon Doyle, HPKB Movement Analysis Ontology, February 12, 1998.
7. Paul Cohen, Robert Schrag, Eric Jones, Adam Pease, Albert Lin, Barbara Starr, David Gunning, and Murray Burke, The DARPA High-Performance Knowledge Bases Project, AI Magazine, Vol. 19, No. 4 (Winter 1998).
8. Christine L. Tsien, TrendFinder: Automated Detection of Alarmable Trends, MIT Ph.D. dissertation, April 28, 2000.
9. Mary T. DeSouza, Automated Medical Trend Detection, MIT M.Eng. thesis, May 22, 2000.
10. Peter Szolovits, Map-based road computations, May 20, 1998. Sources for a Java program that reads maps in the movement analysis challenge problem format and uses an efficient algorithm to find the nearest road link to a given latitude-longitude point (together with its distance to that point). The algorithm uses a KD-tree to compute answers in time logarithmic in the number of nodes in the road network.

**MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)**

The advancement and application of Information Systems Science and Technology to meet Air Force unique requirements for Information Dominance and its transition to aerospace systems to meet Air Force needs.